Requirements Testing

Introduction

In this document we will be evaluating whether the application we designed met the original requirements which were set out in the requirements document, paying most attention to the use cases outlined in the original document. The primary aspect of the use cases we will be covering booking and cancelling lessons, buying product or handling cart actions and registering as a new user.

Booking Lessons

The first use case describes the process of booking a lesson, in which the user must first select the day they wish to book the lesson, then select the lesson at the time they want to book or select to see more details about each lesson. The requirements were met or changed in the following ways:

- Users will select the day they want to book on.
  This has been implemented with buttons for each day.

- The system will request to search for available on that day, the database will then return the results.
  The lesson search has been implemented depending on the day selected, however, available lessons have been stored in session to reduce database communications, reducing the workload for the application. The search has now been implemented in a method within the PHP.

- The list of lessons will be displayed to the user and the user will select the lesson they want to book.
  This has been implemented with a method to generate each lesson. Each lesson also has two buttons, one for booking and once for getting more info.

- The system then checks if they are logged in and instructs them to if they are not.
  This is done by checking if the user variable has been set in sessions, if not they are redirected to log in.

- The system makes a request to see if see if the user has booked any other lessons at this time, returns this lesson if there is one and informs the user, asking them if they still want to book the new lesson.
  This has become redundant as lessons don't have any time overlap and if a user has booked a lesson they are no longer available to be booked again

- The system then sends a message to the database to update who is attending the lesson.

The booked lesson is returned to the database and added to the booked lessons array in session, to avoid having to rest the array with a new database query.

- The system displays a confirmation message to the user and gives them the option to return to the home page or continue in the booking page.
  The confirmation message has been replaced with a message to confirm that the user wants to order the lesson in adherence to the recoverability section of the user requirements. With this implemented the confirmation message was made redundant.

In conclusion, the requirements for this use case were adequately met, with some changes to database operations to reduce the workload of the application and allow for better performance and a change from a confirmation message to a message to confirm they want to book to adhere to UI requirements.

Cancelling Lessons

The use case for cancelling a booking involves the user navigating to the booking page and selecting either to view a booked lessons information or to cancel a booking. The requirements were met or changed in the following ways:

- User selects the booked lesson page.
  The button for this is at the top of the available lessons page.

- The system check if they are logged in and instructs them to if they are not. The system makes a request to see all the lessons booked by the user. The database returns the booked lessons of the user.
  If the user is not logged in they are redirected to the log in page. When the user logs in, a database query is sent to get the booked lessons of that user, which are stored sessions. When redirected to the booked lessons page, the booked lessons are then taken form sessions.

- The system displays the lessons booked by the user.
  This has been implemented with a method to generate the booked lessons with 2 buttons, a button to view lesson info and to delete the lesson.

- The user selects to cancel the booking. The system sends a warning message to the user and asks if they want to continue.
  A popup message has been implemented to allow the user to confirm booking cancelations

- A message is sent to the database to delete that lessons from booking.
  Along with this database query, the booked lesson is also removed from the sessions array.

- The system informs the user that their booking has been cancelled. The system displays the booked lessons of the user, without the deleted booking

The system message here has been removed since the user has already confirmed they want to cancel the booking. This additional message is redundant and impacts the minimal clicks user interface principal outlined in the brief . The page is then reloaded to display the new booked lessons array.

In conclusion, the requirements for this use case were adequately met, with some changes to database operations to reduce the workload of the application and allow for better performance, as well as a change to the messages displayed to adhere to the minimal clicks user interface requirements.

Buying

The use case for Buying involves the user navigating to the Shop page and selecting the product/s they want to order. The requirements were met or changed in the following ways:

• The user selects the Online Store section.

The Store can be selected from the nav bar at the top of the page.

• They are presented with many products and essentials available for purchase.

Users are presented with an array of products available for purchase. The system retrieves all items for sale from the database, including details like price and description.

• The system makes a request to see all the items for sale.

A database query is sent to get the products, which are stored in the database.

• The database returns the items for sale.

The database returns a list of all items which includes essential details for each product.

• The system displays the items for sale.

The system displays the products on the Store page which allows users to browse through them.

• The user then can select a specific product.

This hasn't been implemented.

• After selection, the system makes a request to see detailed information about the specific product.

The individual selection wasn't implemented however detailed information about each product is requested.

• The database returns the information such as details including price, description, available variants for sale.

The database returns a list of all items which includes essential details for each product.

• The system displays the information of the item that is for sale.

The system doesn't display information individually however the information is still displayed but for all products.

• If the Gym Member decides to purchase the product they can proceed directly to checkout pressing the Buy button.

The system checks if they are logged in if they are logged in they can check out.

• The system prompts the Gym Member to confirm the purchase.

The user is prompted if they want to proceed with the checkout.

• If the Gym Member is logged in, the system makes a request to the database to see inventory levels. Then the database processes the request.

Inventory levels were not implemented.

• If the Gym Member is not logged in, they are forced to fill out the shipping details page.

The user is redirected to the register page and the shipping details page has been abandoned because users provide all their details through registration.

• After completing the shipping details, the system makes a request to the database to store the provided information.

Shipping details page is not implemented.

• Then the system makes a request to the database to see inventory levels. Then the database processes the request.

Inventory levels are not implemented.

• Once the purchase is confirmed, the system generates a transaction record and stores it in the database.

Once the purchase is confirmed and all checks are complete the system generates an order record and stores it in the database.

• The Gym Member receives a confirmation message indicating successful purchase along with an order summary.

The Gym Member gets a Thank you for your order! Message and the users can see all of their orders.

• The inventory levels of the item in the database are updated.

The inventory levels of items weren't implemented.

In conclusion, the requirements for this use case were adequately met with a focus on making it easy and efficient for Gym Members to buy what they need. While some features like inventory levels and individual product displays were not included the system does well in showing products and processing orders smoothly. This creates a simple user friendly setup. Overall it meets the main needs and sets a good base for future improvements to make it even easier and smoother for users.

Register & Login/Logout

The user experience is an important factor in our project. Throughout development, we made sure every aspect of the original user specifications were ‘translated’ into a system that works just the way it was envisioned by us. This means the final product offers the functionalities and workflows you expected from the very beginning.

User Registration

- Comprehensive Input: Users can provide all required details (email, first & last name, date of birth, eircode, phone number) and create a unique password, ensuring a complete profile.
- Submission Trigger: The "Register" button initiates the registration process, signalling the system to take action by saving all registration credentials into the database.
- Secure Data Transfer: User registration data is securely transmitted to the database, safeguarding sensitive information.
- Database Confirmation: The database acknowledges successful registration, providing feedback to the system for user notification by being redirected to their profile.

- Persistent Storage: Registration details are stored in the database, allowing users to log in seamlessly with their chosen email and password.
- Success Message: The system provides a "successful registration" message, giving the user clear confirmation.

Administrator Login

- Dedicated Credentials: Administrators have separate usernames and passwords, ensuring controlled access to the management area.
- Authentication Request: The system sends an authentication request to the database, validating administrator login attempts.
- Credential Verification: The database carefully checks provided credentials, guaranteeing only authorized individuals can access the workspace.
- Workspace Access: Upon successful authentication, the administrator is granted access to their workspace, allowing them to perform management tasks.

Login

- Access Control: The login process is the gateway to the system. It helps control who can access the gym management features and any user-specific data.
- Authentication: Verifying a user's email and password against the database is the core security measure. This ensures that only legitimate users can log in.
- Session Management: Upon successful login, a session is likely started to track the user's state. This allows for a seamless experience without needing to re-authenticate on every page.

Conclusion

GymGo’s management system has been carefully crafted to deliver a secure and intuitive experience for both users and administrators. The registration process ensures comprehensive user profiles while prioritizing data security. Login mechanisms provide robust access control, with authentication safeguards in place. The system's design demonstrates a commitment to the user experience, ensuring seamless navigation and clear feedback throughout core interactions.

Conclusion

Most of the initial use cases described in initial requirements document, with some minor changes being made to enhance functionality, such as using sessions instead of constant database queries.